



[> home](#) [> about](#) [> feedback](#) [> login](#)

US Patent & Trademark Office



Try the *new* Portal design

Give us your opinion after using it.

## Search Results

Search Results for: **[profile and hardware and instruction and profiling and Dean and hicks]**

Found **31** of **130,565** searched.

Search within Results



[> Advanced Search](#)

[> Search Help/Tips](#)

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results **1 - 20** of **31**    **short listing**

Prev   
 Page

**1**

**2**

Next   
 Page

- 1** **ProfileMe: hardware support for instruction-level profiling on out-of-order processors** 85%

Jeffrey Dean , James E. Hicks , Carl A. Waldspurger , William E. Weihl , George Chrysos  
**Proceedings of the 30th annual ACM/IEEE international symposium on Microarchitecture** December 1997

Profile data is valuable for identifying performance bottlenecks and guiding optimizations. Periodic sampling of a processor's performance monitoring hardware is an effective, unobtrusive way to obtain detailed profiles. Unfortunately, existing hardware simply counts events, such as cache misses and branch mispredictions, and cannot accurately attribute these events to instructions, especially on out-of-order machines. We propose an alternative approach, called ProfileMe, that samples instructio ...

- 2** **Microprocessor architecture: Frequent loop detection using efficient non-intrusive on-chip hardware** 80%

Ann Gordon-Ross , Frank Vahid  
**Proceedings of the international conference on Compilers, architectures and synthesis for embedded systems** October 2003

Dynamic software optimization methods are becoming increasingly popular for improving software performance and power. The first step in dynamic optimization consists of detecting frequently executed code, or "critical regions." Previous critical region detectors have been targeted to desktop processors. We introduce a critical region detector targeted to embedded processors, with the unique features of being very size and power efficient, and being completely non-intrusive to the software's exec ...

- 3** **Rapid profiling via stratified sampling** 80%



S. Subramanya Sastry , Rastislav Bodík , James E. Smith  
**ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on Computer architecture** May 2001  
 Volume 29 Issue 2

*Sophisticated binary translators and dynamic optimizers demand a program profiler with low overhead, high accuracy, and the ability to collect a variety of profile types. A profiling scheme that achieves these goals is proposed. Conceptually, the hardware compresses a stream of profile data by counting identical events; the compressed profile data is passed to software for analysis. Compressing the high-bandwidth event stream greatly reduces software overhead. Because optimizations can tolerate ...*

#### 4 Cache decay: exploiting generational behavior to reduce cache leakage 80%



power  
 Stefanos Kaxiras , Zhigang Hu , Margaret Martonosi  
**ACM SIGARCH Computer Architecture News , Proceedings of the 28th annual international symposium on Computer architecture** May 2001  
 Volume 29 Issue 2

*Power dissipation is increasingly important in CPUs ranging from those intended for mobile use, all the way up to high-performance processors for high-end servers. While the bulk of the power dissipated is dynamic switching power, leakage power is also beginning to be a concern. Chipmakers expect that in future chip generations, leakage's proportion of total chip power will increase significantly.*

*This paper examines methods for reducing leakage power within the cache memory ...*

#### 5 Probing the black box: User-level internet path diagnosis 80%



Ratul Mahajan , Neil Spring , David Wetherall , Thomas Anderson  
**Proceedings of the nineteenth ACM symposium on Operating systems principles**  
 October 2003





Diagnosing faults in the Internet is arduous and time-consuming, in part because the network is composed of diverse components spread across many administrative domains. We consider an extreme form of this problem: can end users, with no special privileges, identify and pinpoint faults inside the network that degrade the performance of their applications? To answer this question, we present both an architecture for user-level Internet path diagnosis and a practical tool to diagnose paths in the ...

#### 6 DiST: a simple, reliable and scalable method to significantly reduce 80%



processor architecture simulation time  
 Sylvain Girbal , Gilles Mouchard , Albert Cohen , Olivier Temam  
**ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems** June 2003  
 Volume 31 Issue 1

While architecture simulation is often treated as a methodology issue, it is at the core of most processor architecture research works, and simulation speed is often the bottleneck of the typical trial-and-error research process. To speedup simulation during this research process and get trends faster, researchers usually reduce the trace size. More sophisticated techniques like trace sampling or distributed simulation are scarcely used because they are considered unreliable and complex due to t ...

- 7** Profiling tools for hardware/software partitioning of embedded applications 80%  
 Dinesh C. Suresh , Walid A. Najjar , Frank Vahid , Jason R. Villarreal , Greg Stitt  
**ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems** June 2003  
Volume 38 Issue 7  
Loops constitute the most executed segments of programs and therefore are the best candidates for hardware software partitioning. We present a set of profiling tools that are specifically dedicated to loop profiling and do support combined function and loop profiling. One tool relies on an instruction set simulator and can therefore be augmented with architecture and micro-architecture features simulation while the other is based on compile-time instrumentation of gcc and therefore has very litt ...
- 8** Scalable analysis techniques for microprocessor performance counter metrics 80%  
 Dong H. Ahn , Jeffrey S. Vetter  
**Proceedings of the 2002 ACM/IEEE conference on Supercomputing** November 2002  
Contemporary microprocessors provide a rich set of integrated performance counters that allow application developers and system architects alike the opportunity to gather important information about workload behaviors. Current techniques for analyzing data produced from these counters use raw counts, ratios, and visualization techniques help users make decisions about their application performance. While these techniques are appropriate for analyzing data from one process, they do not scale easi ...
- 9** Compilation and run-time systems: Vacuum packing: extracting hardware-detected program phases for post-link optimization 80%  
 Ronald D. Barnes , Erik M. Nystrom , Matthew C. Merten , Wen-mei W. Hwu  
**Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture** November 2002  
This paper presents Vacuum Packing, a new approach to profile-based program optimization. Instead of using traditional aggregate or summarized execution profile weights, this approach uses a transparent hardware profiler to automatically detect execution phases and record branch profile information for each new phase. The code extraction algorithm then produces code packages that are specially formed for their corresponding phases. The algorithm compensates for the incomplete and often incoheren ...
- 10** Efficient instrumentation for code coverage testing 80%  
 Mustafa M. Tikir , Jeffrey K. Hollingsworth  
**ACM SIGSOFT Software Engineering Notes , Proceedings of the international symposium on Software testing and analysis** July 2002  
Volume 27 Issue 4  
Evaluation of Code Coverage is the problem of identifying the parts of a program that did not execute in one or more runs of a program. The traditional approach for code coverage tools is to use static code instrumentation. In this paper we present a new approach to dynamically insert and remove instrumentation code to reduce the runtime overhead of code coverage. We also explore the use of dominator tree information to reduce the number of instrumentation points needed. Our experiments show tha ...

- 11** Exploiting prolific types for memory management and optimizations 80%



Yefim Shuf , Manish Gupta , Rajesh Bordawekar , Jaswinder Pal Singh  
**ACM SIGPLAN Notices , Proceedings of the 29th ACM SIGPLAN-SIGACT symposium on Principles of programming languages** January 2002  
 Volume 37 Issue 1

In this paper, we introduce the notion of *prolific* and *non-prolific* types, based on the number of instantiated objects of those types. We demonstrate that distinguishing between these types enables a new class of techniques for memory management and data locality, and facilitates the deployment of known techniques. Specifically, we first present a new *type-based* approach to garbage collection that has similar attributes but lower cost than generational collection. Then we de ...

## 12 Efficient and flexible value sampling

80%



M. Burrows , U. Erlingson , S-T. A. Leung , M. T. Vandevoorde , C. A. Waldspurger , K. Walker , W. E. Weihl  
**Proceedings of the ninth international conference on Architectural support for programming languages and operating systems** November 2000  
 Volume 34 , 28 Issue 5 , 5

This paper presents novel sampling-based techniques for collecting statistical profiles of register contents, data values, and other information associated with instructions, such as memory latencies. Values of interest are sampled in response to periodic interrupts. The resulting value profiles can be analyzed by programmers and optimizers to improve the performance of production uniprocessor and multiprocessor systems. Our value sampling system extends the DCPI continuous profiling infrastru ...

## 13 Tools for application-oriented performance tuning

80%



John Mellor-Crummey , Robert Fowler , David Whalley  
**Proceedings of the 15th international conference on Supercomputing** June 2001

Application performance tuning is a complex process that requires assembling various types of information and correlating it with source code to pinpoint the causes of performance bottlenecks. Existing performance tools don't adequately support this process in one or more dimensions. We discuss some of the critical utility and usability issues for application-level performance analysis tools in the context of two performance tools, *MHSim* and *HPCView*, that we built to support our ...

## 14 Efficient and flexible value sampling

80%



M. Burrows , U. Erlingson , S.-T. A. Leung , M. T. Vandevoorde , C. A. Waldspurger , K. Walker , W. E. Weihl  
**ACM SIGPLAN Notices** November 2000  
 Volume 35 Issue 11

This paper presents novel sampling-based techniques for collecting statistical profiles of register contents, data values, and other information associated with instructions, such as memory latencies. Values of interest are sampled in response to periodic interrupts. The resulting value profiles can be analyzed by programmers and optimizers to improve the performance of production uniprocessor and multiprocessor systems. Our value sampling system extends the DCPI continuous profiling infrastru ...

## 15 Automated data-member layout of heap objects to improve memory-hierarchy performance

80%



Thomas Kistler , Michael Franz  
**ACM Transactions on Programming Languages and Systems (TOPLAS)** May 2000

## Volume 22 Issue 3

We present and evaluate a simple, yet efficient optimization technique that improves memory-hierarchy performance for pointer-centric applications by up to 24% and reduces cache misses by up to 35%. This is achieved by selecting an improved ordering for the data members of pointer-based data structures. Our optimization is applicable to all type-safe programming languages that completely abstract from physical storage layout; examples of such languages are Java and Oberon. Our technique doe ...

**16 A portable sampling-based profiler for Java virtual machines**

80%



John Whaley

**Proceedings of the ACM 2000 conference on Java Grande** June 2000**17 Performance analysis of the Alpha 21264-based Compaq ES40 system**

80%



Zarka Cvetanovic , R. E. Kessler

**ACM SIGARCH Computer Architecture News , Proceedings of the 27th annual international symposium on Computer architecture** May 2000

## Volume 28 Issue 2

This paper evaluates performance characteristics of the Compaq ES40 shared memory multiprocessor. The ES40 system contains up to four Alpha 21264 CPU's together with a high-performance memory system. We qualitatively describe architectural features included in the 21264 microprocessor and the surrounding system chipset. We further quantitatively show the performance effects of these features using benchmark results and profiling data collected from industry-standard commercial and t ...

**18 A hardware-driven profiling scheme for identifying program hot spots to support runtime optimization**

80%



Matthew C. Merten , Andrew R. Trick , Christopher N. George , John C. Gyllenhaal , Wenmei W. Hwu

**ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture** May 1999

## Volume 27 Issue 2

This paper presents a novel hardware-based approach for identifying, profiling, and monitoring hot spots in order to support runtime optimization of general purpose programs. The proposed approach consists of a set of tightly coupled hardware tables and control logic modules that are placed in the retirement stage of a processor pipeline removed from the critical path. The features of the proposed design include rapid detection of program hot spots after changes in execution behavior, runtime-tu ...

**19 Variable length path branch prediction**

80%



Jared Stark , Marius Evers , Yale N. Patt

**Proceedings of the eighth international conference on Architectural support for programming languages and operating systems** October 1998

## Volume 33 , 32 Issue 11 , 5

Accurate branch prediction is required to achieve high performance in deeply pipelined, wide-issue processors. Recent studies have shown that conditional and indirect (or computed) branch targets can be accurately predicted by recording the path, which consists of the target addresses of recent branches, leading up to the branch. In current path based branch predictors, the  $N$  most recent target addresses are hashed together to form an index into a table, where  $N$  is some fixed intege ...

80%

**20 JRes: a resource accounting interface for Java**

Grzegorz Czajkowski , Thorsten von Eicken

**ACM SIGPLAN Notices , Proceedings of the 13th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications** October 1998

Volume 33 Issue 10

With the spread of the Internet the computing model on server systems is undergoing several important changes. Recent research ideas concerning dynamic operating system extensibility are finding their way into the commercial domain, resulting in designs of extensible databases and Web servers. In addition, both ordinary users and service providers must deal with untrusted downloadable executable code of unknown origin and intentions. Across the board, Java has emerged as the language of choice fo ...

---

**Results 1 - 20 of 31      short listing**  
Prev  
Page**1****2**  
Next  
Page

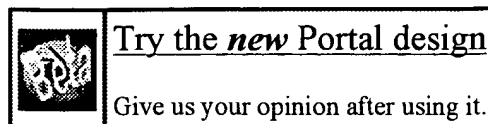
---

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.



[> home](#) [> about](#) [> feedback](#) [> login](#)

US Patent & Trademark Office



## Search Results

Search Results for: **[hardware-based and profiling and without and compiler]**  
Found **132** of **130,565** searched.

## Search within Results



[> Advanced Search](#)

[> Search Help/Tips](#)

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results 1 - 20 of 132 **short listing**



1

2

3

4

5

6

7



### 1 Survey of code-size reduction methods

87%



Árpád Beszédes , Rudolf Ferenc , Tibor Gyimóthy , André Dolenc , Konsta Karsisto

**ACM Computing Surveys (CSUR)** September 2003

Volume 35 Issue 3

Program code compression is an emerging research activity that is having an impact in several production areas such as networking and embedded systems. This is because the reduced-sized code can have a positive impact on network traffic and embedded system costs such as memory requirements and power consumption. Although code-size reduction is a relatively new research area, numerous publications already exist on it. The methods published usually have different motivations and a variety of appli ...

### 2 Accurate and practical profile-driven compilation using the profile buffer 87%



Thomas M. Conte , Kishore N. Menezes , Mary Ann Hirsch

**Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture** December 1996

Profiling is a technique of gathering program statistics in order to aid program optimization. In particular, it is an essential component of compiler optimization for the extraction of instruction-level parallelism. Code instrumentation has been the most popular method of profiling. However, real-time, interactive, and transaction processing applications suffer from the high execution-time overhead imposed by software instrumentation. This paper suggests the use of hardware dedicated to the tas ...

### 3 Design and evaluation of a compiler algorithm for prefetching

87%



Todd C. Mowry , Monica S. Lam , Anoop Gupta

**ACM SIGPLAN Notices , Proceedings of the fifth international conference on Architectural support for programming languages and operating systems**

Search Forms

Search Results

Help

User Searches

Preferences

Logout

# Refine Search

## Search Results -

Terms	Documents
(712/298  712/215).ccls.	384

Database:

US Pre-Grant Publication Full-Text Database  
 US Patents Full-Text Database  
 US OCR Full-Text Database  
 EPO Abstracts Database  
 JPO Abstracts Database  
 Derwent World Patents Index  
 IBM Technical Disclosure Bulletins

Search:

L13

Refine Search

Recall Text

Clear

Interrupt

## Search History

DATE: Sunday, April 11, 2004 [Printable Copy](#) [Create Case](#)

### Set Name Query

side by side

### Hit Count Set Name

result set

DB=USPT; PLUR=YES; OP=ADJ

L13 712/298,215.ccls.

384 L13

L12 717/151.ccls.

176 L12

L11 717/127,128,129,130,131,154,158.ccls.

900 L11

DB=TDBD; PLUR=YES; OP=ADJ

L10 (hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)

0 L10

DB=DWPI; PLUR=YES; OP=ADJ

L9 (hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)

0 L9

DB=JPAB; PLUR=YES; OP=ADJ

L8 (hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)

0 L8

DB=EPAB; PLUR=YES; OP=ADJ

L7 (hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)

0 L7

DB=USOC; PLUR=YES; OP=ADJ

L6 (hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)

0 L6

DB=PGPB; PLUR=YES; OP=ADJ



<u>L5</u>	(hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)	22	<u>L5</u>
<i>DB=USPT; PLUR=YES; OP=ADJ</i>			
<u>L4</u>	(hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)	17	<u>L4</u>
<u>L3</u>	L2 and hardware\$ and software\$ and compil\$	1	<u>L3</u>
<u>L2</u>	5355487.pn.	1	<u>L2</u>
<u>L1</u>	(hardware near5 profil\$) and ((without\$ or no\$) near5 compil\$)	17	<u>L1</u>

END OF SEARCH HISTORY